REMARKS

This is in response to the Final Office Action dated April 7, 2005 in which claims 28-55 were rejected under 35 U.S.C. § 102 as being anticipated by Template Software. With this Amendment, claims 28, 43 and 49 are amended. Claims 28-55 are pending in this application.

Claim Correction

It is respectfully requested that the amendment to claim 43 be entered to correct the spelling of the word "architecture."

Claim Rejections Under 35 U.S.C. § 102

Claim 28 is amended to make a number of clarifications. First, the claim amendment clarifies that the components generated by the component development tool are "autonomous." Second, the claim clarifies that the specified features of the system development tool do not require writing of any code. Third, the claim clarifies that the engine software program enables communication between component instances based upon the defined links.

With this Amendment, claim 28 is allowable over the prior art of record. The first element of amended claim 28 recites a component development tool for generating a plurality of autonomous components that implement and consume services. Autonomous components of the present invention are described, for example, at page 8, lines 9-15 (with reference to FIG. 3), which explains that the components of the present invention can "interact while being self-sufficient," and that components are "designed to operate as stand-alone entities, with no knowledge of peers or proxies, executing their own activities."

Template software does not disclose a tool for generating autonomous components. Rather, Template Software's approach is to generate code to define all aspects of the desired system and then compiling the code into a monolithic application. Evidence of the compiled nature of the system is found on page 8-20 of the document "Using the WFT Development Environment," where it says "You must build a business process node or server before you can run it." "Build" is defined on page 8-3 of the

same document as "To compile the source files ... into object files that can be executed. Building a business process node or server is the same as building an application that runs in the node or server." As a result, it can be seen that Template Software does not utilize autonomous components, because all relationships in the system are compiled. Rather, the complete system model must be defined up-front before the build. Furthermore, any subsequent changes to the system require a complete re-build and redeployment of the system. Therefore, Template Software does not disclose autonomous components as recited by claim 28. As a result, claim 28 is allowable over the prior art of record.

The second element of amended independent claim 28 recites a system development tool for defining a plurality of component instances based on the plurality of components, configuring the plurality of component instances, and defining links between component instances, without requiring writing of code. The amendment clarifies that the system development tool does not require writing of code. Although a developer must write code and compile the code to create a component, code need not be written in the system development tool to utilize the component as defined in the claim.

In discussing the second element of independent claim 28, the Office Action stated that defining component instances, configuring component instances, and defining links between component instances, all without requiring writing of additional code were "basic concepts of object-oriented technology." This is not correct–the features of the present invention are not merely basic concepts of object-oriented technology. First, the Office Action states that defining component instances is the same as defining an object. However, the act of defining an object requires writing at least one line of code to define the object. Second, the Office Action states that configuring component instances is the same as writing methods that get and set values. However, writing methods is also writing code. Third, the Office Action states that defining links between component instances is one of the inherent aspects of object technology, namely messaging. However, messaging between objects also requires writing of code. Template Software's messaging between objects is described in more detail below. Since each of these features is done through the writing of code in the Template Software system, Template Software does not

teach or suggest a system development tool for defining component instances, configuring component instances, and defining links between component instances without requiring writing of code.

The third element of amended independent claim 28 recites an engine software program to provide a dynamic run-time environment for hosting the plurality of component instances and supporting communication between component instances based on the defined links. The amendment clarifies that communication between component instances is based on the defined links. This feature of the present invention is related to the previously described feature of having autonomous components. Since the components are autonomous, they are stand-alone entities, with no knowledge of peers or proxies. (P. 8, lines 9-15.) When a component is created the developer defines the messages that the component is capable of receiving on its incoming ports. The developer then defines the logic as to how to respond to those incoming messages, and defines the messages that the component will send out of its outgoing ports. The component is compiled without defining the source of incoming messages or the recipients of outgoing messages. Links, which are uncompiled relationships between components can be established or modified at any time while the component is running, without components being aware of the existence of those links. A link that is defined between two components means that a message coming from the output port of one component will be delivered to the incoming port of another component, without either component having any knowledge of the other. The message is simply delivered, without the source expecting a response. Components can exist without any links to other components, although in this case they will have no incoming messages. Un-compiled relationships between autonomous components allows a high level of flexibility as these relationships can be established, modified or deleted at any time. Additional components can be introduced into the system at any time, while the system is running. These new components may be linked to any other compatible component in the system, providing extensibility/modification to the system functionality on the fly.

Template Software does not disclose an engine software program to provide a dynamic run-time environment for hosting the plurality of component instances and supporting communication

between component instances based on the defined links. Rather, as described above, Template Software uses a method of generating code to define all aspects of the desired system and then compiling the code into a monolithic application. Since Template Software does not teach each and every element of claim 28, claim 28 is in condition for allowance. Reconsideration and notice to that effect is respectfully requested.

With this Amendment, independent claim 49 also is amended to clarify that components are autonomous. This feature of the present invention, as described above, is not disclosed by Template Software. Based upon this and other features defined in the claim and explained above that are not disclosed by Template Software, independent claim 49 is in condition for allowance. Reconsideration and notice to that effect is respectfully requested.

Dependent claims 29-48 and 50-55 depend from allowable independent claims 28 and 49 respectfully and are therefore allowable.

Conclusion

In view of the foregoing, this application containing pending claims 28-55 is in condition for allowance. Reconsideration and notice to that effect is respectfully requested.

Respectfully submitted,

KINNEY & LANGE, P.A.

Date: 6/7/05 By: _____

David R. Fairbairn, Reg. No. 26,047
THE KINNEY & LANGE BUILDING
312 South Third Street
Minneapolis, MN 55415-1002
Telephone: (612) 339-1863
Fax: (612) 339-6580

DRF:BAT:bmg